```
H 13
```

```
FFFFFFFFFFFFFFF     AAAAAAAAA     LLL
FFFFFFFFFFFFFFF     AAAAAAAAA     LLL
FFFFFFFFFFFFFFF     AAAAAAAAA     LLL
FFF              AAA         AAA  LLL
FFF              AAA         AAA  LLL
FFF              AAA         AAA  LLL
FFF              AAA         AAA  LLL
FFF              AAA         AAA  LLL
FFF              AAA         AAA  LLL
FFFFFFFFFFFF     AAA         AAA  LLL
FFFFFFFFFFFF     AAA         AAA  LLL
FFFFFFFFFFFF     AAA         AAA  LLL
FFF              AAAAAAAAAAAAAAA  LLL
FFF              AAAAAAAAAAAAAAA  LLL
FFF              AAAAAAAAAAAAAAA  LLL
FFF              AAA         AAA  LLL
FFF              AAA         AAA  LLL
FFF              AAA         AAA  LLL
FFF              AAA         AAA  LLLLLLLLLLLLLLL
FFF              AAA         AAA  LLLLLLLLLLLLLLL
FFF              AAA         AAA  LLLLLLLLLLLLLLL
```

```
_$

Val
---
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
```

```
**FILE**ID**FALBLDXAB                                    H 14

FFFFFFFFFF    AAAAAA    LL          BBBBBBBB    LL          DDDDDDDD    XX      XX    AAAAAA    BBBBBBBB
FFFFFFFFFF    AAAAAA    LL          BBBBBBBB    LL          DDDDDDDD    XX      XX    AAAAAA    BBBBBBBB
FF          AA      AA  LL          BB      BB  LL          DD      DD  XX      XX  AA      AA  BB      BB
FF          AA      AA  LL          BB      BB  LL          DD      DD  XX      XX  AA      AA  BB      BB
FF          AA      AA  LL          BB      BB  LL          DD      DD    XX  XX    AA      AA  BB      BB
FF          AA      AA  LL          BB      BB  LL          DD      DD    XX  XX    AA      AA  BB      BB
FFFFFFFF    AA      AA  LL          BBBBBBBB    LL          DD      DD      XX      AA      AA  BBBBBBBB
FFFFFFFF    AA      AA  LL          BBBBBBBB    LL          DD      DD      XX      AA      AA  BBBBBBBB
FF          AAAAAAAAAA  LL          BB      BB  LL          DD      DD    XX  XX    AAAAAAAAAA  BB      BB
FF          AAAAAAAAAA  LL          BB      BB  LL          DD      DD    XX  XX    AAAAAAAAAA  BB      BB
FF          AA      AA  LL          BB      BB  LL          DD      DD  XX      XX  AA      AA  BB      BB      ....
FF          AA      AA  LL          BB      BB  LL          DD      DD  XX      XX  AA      AA  BB      BB      ....
FF          AA      AA  LLLLLLLLLL  BBBBBBBB    LLLLLLLLLL  DDDDDDDD    XX      XX  AA      AA  BBBBBBBB        ....
FF          AA      AA  LLLLLLLLLL  BBBBBBBB    LLLLLLLLLL  DDDDDDDD    XX      XX  AA      AA  BBBBBBBB        ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II        SS
LL              II        SS
LL              II        SS
LL              II        SS
LL              II          SSSSSS
LL              II          SSSSSS
LL              II                SS
LL              II                SS
LL              II                SS
LL              II                SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

FAL
V04

Table of contents

```
0000          .TITLE  FALBLDXAB - BUILD DAP EXT ATT MESSAGES
0000          .IDENT  'V04-000'
0000
0000    ;
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000    ;
0000
0000    ;++
0000    ; Facility: FAL (DECnet File Access Listener)
0000    ;
0000    ; Abstract: This module builds the DAP extended Attributes messages.
0000    ;
0000    ; Environment: VAX/VMS, user mode
0000    ;
0000    ; Author: James A. Krycka,      Creation Date:  22-MAY-1979
0000    ;
0000    ; Modified By:
0000    ;
0000    ;       V03-002 JAK0136         J A Krycka      07-MAR-1984
0000    ;               Cleanup.
0000    ;
0000    ;--
```

```
0000          .SBTTL  DECLARATIONS
0000
0000    ;
0000    ; Include Files:
0000    ;
0000
0000          $DAPPLGDEF                           ; Define DAP prologue symbols
0000          $DAPHDRDEF                           ; Define DAP message header
0000          $DAPATTDEF                           ; Define DAP Attributes message
0000          $DAPKEYDEF                           ; Define DAP Key Definition message
0000          $DAPALLDEF                           ; Define DAP Allocation message
0000          $DAPSUMDEF                           ; Define DAP Summary message
0000          $DAPTIMDEF                           ; Define DAP Date and Time message
0000          $DAPPRODEF                           ; Define DAP Protection message
0000          $FALWRKDEF                           ; Define FAL Work Area symbols
0000          $FABDEF                              ; Define File Access Block symbols
0000          $XABDEF                              ; Define symbols common to all XABs
0000          $XABALLDEF                           ; Define Allocation XAB symbols
0000          $XABDATDEF                           ; Define Date and Time XAB symbols
0000          $XABKEYDEF                           ; Define Key Definition XAB symbols
0000          $XABPRODEF                           ; Define Protection XAB symbols
0000          $XABSUMDEF                           ; Define Summary XAB symobls
0000
0000    ;
0000    ; Macros:
0000    ;
0000    ;       None
0000    ;
0000    ; Equated Symbols:
0000    ;
0000
0000          ASSUME  DAP$Q_DCODE_FLG EQ 0
0000          ASSUME  FAL$Q_FLG EQ 0
0000
0000    ;
0000    ; Own Storage:
0000    ;
```

```
                              0000     83         .SBTTL  FAL$ENCODE_KEY
                          00000000     84         .PSECT  FAL$CODE          NOSHR,EXE,RD,NOWRT,BYTE
                              0000     85
                              0000     86 ;++
                              0000     87 ; Functional Description:
                              0000     88 ;
                              0000     89 ;       FAL$ENCODE_KEY builds the specified DAP Key Definition message.
                              0000     90 ;
                              0000     91 ; Calling Sequence:
                              0000     92 ;
                              0000     93 ;       BSBW    FAL$ENCODE_KEY
                              0000     94 ;
                              0000     95 ; Input Parameters:
                              0000     96 ;
                              0000     97 ;       R6      Key of reference value
                              0000     98 ;       R8      Address of FAL work area
                              0000     99 ;       R9      Address of DAP control block
                              0000    100 ;       R10     Address of FAB
                              0000    101 ;       R11     Address of RAB
                              0000    102 ;
                              0000    103 ; Implicit Inputs:
                              0000    104 ;
                              0000    105 ;       FAB$B_ORG
                              0000    106 ;
                              0000    107 ; Output Parameters:
                              0000    108 ;
                              0000    109 ;       R0-R6   Destroyed
                              0000    110 ;       R7      Address of XAB
                              0000    111 ;
                              0000    112 ; Implicit Outputs:
                              0000    113 ;
                              0000    114 ;       None
                              0000    115 ;
                              0000    116 ; Completion Codes:
                              0000    117 ;
                              0000    118 ;       None
                              0000    119 ;
                              0000    120 ; Side Effects:
                              0000    121 ;
                              0000    122 ;       None
                              0000    123 ;
                              0000    124 ;--
                              0000    125
                              0000    126 FAL$ENCODE_KEY::                              ; Entry point
56   0000004C 8F        C4    0000    127         MULL2   #FAL$K_KEYXAB,R6              ; Using REF as an index, compute
  57     1000 C846      9E    0007    128         MOVAB   FAL$L_KEYXAB(R8)[R6],R7       ;  address of Key Definition XAB to use
         50   0A        D0    000D    129         MOVL    #DAP$K_KEY_MSG,R0             ; Get message type value
            FFED'       30    0010    130         BSBW    FAL$BUILD_HEAD                ; Construct message header
      20   1D AA        91    0013    131         CMPB    FAB$B_ORG(R10),#FAB$C_IDX     ; Build dummy message (all fields
                 03     13    0017    132         BEQL    5$                            ;  defaulted) if file ORG is not IDX
               0097     31    0019    133         BRW     40$                           ; Branch aid
51   0007EFFF 8F        D0    001C    134 5$:     MOVL    #<DAP$M_FLG!-                 ; Get key menu value
                              0023    135                   DAP$M_DFL!-                 ;
                              0023    136                   DAP$M_IFL!-                 ;
                              0023    137                   DAP$M_NSG!-                 ;
                              0023    138                   DAP$M_REF!-                 ;
                              0023    139                   DAP$M_KNM!-                 ;
```

```
FAL
Sym

DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
DAF
```

```
                        0023                      DAP$M_NUL!-             ;
                        0023                      DAP$M_IAN!-             ;
                        0023                      DAP$M_LAN!-             ;
                        0023                      DAP$M_DAN!-             ;
                        0023                      DAP$M_DTP!-             ;
                        0023                      DAP$M_RVB!-             ;
                        0023                      DAP$M_DVB!-             ;
                        0023                      DAP$M_DBS!-             ;
                        0023                      DAP$M_IBS!-             ;
                        0023                      DAP$M_LVL!-             ;
                        0023                      DAP$M_TKS!-             ;
                        0023                      DAP$M_MRL!-             ;
                        0023                      0>,R1                   ;
           FFDA'   30   0023            BSBW      FAL$CVT_BN4_EXT         ; Store KEYMENU as an extensible field
                        0026
                        0026    ;
                        0026    ; Include the FLG, DFL, and IFL fields in the message.
                        0026    ;
                        0026
    51    12 A7    9A   0026            MOVZBL    XAB$B_FLG(R7),R1        ; Get FLG bits returned by RMS
             52    D4   002A            CLRL      R2                      ; Clear corresponding DAP bits
                        002C            $MAPBIT XAB$V_DUP,DAP$V_DUP       ; Map DUP bit
                        0034            $MAPBIT XAB$V_CHG,DAP$V_CHG       ; Map CHG bit
                        003C            $MAPBIT XAB$V_NUL,DAP$V_NUL_CHR   ; Map NUL bit
       83    52    90   0044            MOVB      R2,(R3)+                ; Store key options as extensible field
    83    1C A7    B0   0047            MOVW      XAB$W_DFL(R7),(R3)+     ; Store data bucket fill quantity field
    83    1A A7    B0   004B            MOVW      XAB$W_IFL(R7),(R3)+     ; Store index bucket fill quantity field
                        004F
                        004F    ;
                        004F    ; Include the NSG, POS, and SIZ fields in the message.
                        004F    ;
                        004F
    50    14 A7    90   004F            MOVB      XAB$B_NSG(R7),R0        ; Get loop count
       83    50    90   0053            MOVB      R0,(R3)+                ; Store number of key segments field
             11    13   0056            BEQL      20$                     ; Branch if zero
    51    1E A7    3E   0058            MOVAW     XAB$W_POS(R7),R1        ; Get address of POS array
    52    2E A7    9E   005C            MOVAB     XAB$B_SIZ(R7),R2        ; Get address of SIZ arrary
       83    81    B0   0060    10$:    MOVW      (R1)+,(R3)+             ; Store key segment position field
       83    82    90   0063            MOVB      (R2)+,(R3)+             ; Store key segment size field
          F7 50    F5   0066            SOBGTR    R0,10$                  ; Loop if more to go
                        0069
                        0069    ;
                        0069    ; Include the REF, KNM, NUL, IAN, LAN, DAN, and DTP fields in the message.
                        0069    ;
                        0069
    83    17 A7    90   0069    20$:    MOVB      XAB$B_REF(R7),(R3)+     ; Store key of reference field
             83    94   006D            CLRB      (R3)+                   ; Assume no key name buffer
          38 A7    D5   006F            TSTL      XAB$L_KNM(R7)           ; Branch if no key name buffer
             09    13   0072            BEQL      30$                     ;
 FF A3       20    90   0074            MOVB      #32,-1(R3)              ; Store KNM as an image field
63    38 B7  20    28   0078            MOVC3     #32,@XAB$L_KNM(R7),(R3) ; Copy 32-byte key name field into msg
    83    15 A7    90   007D    30$:    MOVB      XAB$B_NUL(R7),(R3)+     ; Store null key character field
    83    08 A7    90   0081            MOVB      XAB$B_IAN(R7),(R3)+     ; Store index area number field
    83    09 A7    90   0085            MOVB      XAB$B_LAN(R7),(R3)+     ; Store lowest level index area
                        0089                                              ;  number field
    83    0A A7    90   0089            MOVB      XAB$B_DAN(R7),(R3)+     ; Store data area number field
    83    13 A7    90   008D            MOVB      XAB$B_DTP(R7),(R3)+     ; Store key data type field
```

```
                      0091
                      0091      ;
                      0091      ; Include the RVB, DVB, DBS, IBS, LVL, TKS, and MRL fields in the message.
                      0091      ;
                      0091
51    0E A7    D0     0091                MOVL    XAB$L_RVB(R7),R1        ; Get root bucket start VBN value
        FF68'  30     0095                BSBW    FAL$CVT_BN4_IMG         ; Store RVB as an image field
51    3C A7    D0     0098                MOVL    XAB$L_DVB(R7),R1        ; Get first data bucket start VBN value
        FF61'  30     009C                BSBW    FAL$CVT_BN4_IMG         ; Store DVB as an image field
83    0D A7    90     009F                MOVB    XAB$B_DBS(R7),(R3)+     ; Store data bucket fill size field
83    0C A7    90     00A3                MOVB    XAB$B_IBS(R7),(R3)+     ; Store index bucket fill size field
83    0B A7    90     00A7                MOVB    XAB$B_LVL(R7),(R3)+     ; Store level of root buckets field
83    16 A7    90     00AB                MOVB    XAB$B_TKS(R7),(R3)+     ; Store total key size field
83    18 A7    B0     00AF                MOVW    XAB$W_MRL(R7),(R3)+     ; Store minimum record length to contain
                      00B3                                                ;  key field
        FF4A'  30     00B3      40$:      BSBW    FAL$BUILD_TAIL          ; Finish building message
           05  00B6                       RSB                             ; Exit
```

```
FAL
Pse

PSE
---
.
$AB
FAL

Pha
---
Ini
Com
Pas
Sym
Pas
Sym
Pse
Cro
Ass

The
726
The
614
30

Mac
---
_$2
_$2
TOT

150

The

MAC
```

```
                         00B7          .SBTTL  FAL$ENCODE_ALL
                     000000B7          .PSECT  FAL$CODE          NOSHR,EXE,RD,NOWRT,BYTE
                         00B7
                         00B7     ;++
                         00B7     ; Functional Description:
                         00B7     ;
                         00B7     ;       FAL$ENCODE_ALL builds the specified DAP Allocation message.
                         00B7     ;
                         00B7     ; Calling Sequence:
                         00B7     ;
                         00B7     ;       BSBW    FAL$ENCODE_ALL
                         00B7     ;
                         00B7     ; Input Parameters:
                         00B7     ;
                         00B7     ;       R6      Area ID value
                         00B7     ;       R8      Address of FAL work area
                         00B7     ;       R9      Address of DAP control block
                         00B7     ;       R10     Address of FAB
                         00B7     ;       R11     Address of RAB
                         00B7     ;
                         00B7     ; Implicit Inputs:
                         00B7     ;
                         00B7     ;       DAP$V_VAXVMS
                         00B7     ;
                         00B7     ; Output Parameters:
                         00B7     ;
                         00B7     ;       R0-R6   Destroyed
                         00B7     ;       R7      Address of XAB
                         00B7     ;
                         00B7     ; Implicit Outputs:
                         00B7     ;
                         00B7     ;       None
                         00B7     ;
                         00B7     ; Completion Codes:
                         00B7     ;
                         00B7     ;       None
                         00B7     ;
                         00B7     ; Side Effects:
                         00B7     ;
                         00B7     ;       None
                         00B7     ;
                         00B7     ;--
                         00B7
                         00B7     FAL$ENCODE_ALL::                                  ; Entry point
      56     20     C4   00B7             MULL2   #FAL$K_ALLXAB,R6                  ; Using AID as an index, compute
 57   0C00 C846     9E   00BA             MOVAB   FAL$L_ALLXAB(R8)[R6],R7           ;  address of Allocation XAB to use
      50     0B     D0   00C0             MOVL    #DAP$K_ALL_MSG,R0                 ; Get message type value
          FF3A'     30   00C3             BSBW    FAL$BUILD_HEAD                    ; Construct message header
   51   01E5 8F     3C   00C6             MOVZWL  #<DAP$M_VOL!-                     ; Get allocation menu value
                         00CB                       DAP$M_AOP!-                     ;
                         00CB                       DAP$M_ALQ2!-                    ;
                         00CB                       DAP$M_AID!-                     ;
                         00CB                       DAP$M_BKZ!-                     ;
                         00CB                       DAP$M_DEQ2!-                    ;
                         00CB                     0>,R1                             ;
   03 69     34     E1   00CB             BBC     #DAP$V_VAXVMS,(R9),10$            ; Branch if partner is not VAX/VMS
      51     0A     A8   00CF             BISW2   #<DAP$M_ALN!DAP$M_LOC>,R1 ; Add to menu
```

```
           FF2B'  30  00D2   272 10$:    BSBW    FAL$CVT_BN4_EXT          ; Store ALLMENU as an extensible field
                      00D5   273
                      00D5   274 ;
                      00D5   275 ; Include the VOL, ALN, and AOP fields in the message.
                      00D5   276 ;
                      00D5   277
    83   0A A7    B0  00D5   278         MOVW    XAB$W_VOL(R7),(R3)+      ; Store relative volume number field
    04 69    34   E1  00D9   279         BBC     #DAP$V_VAXVMS,(R9),20$   ; Branch if partner is not VAX/VMS
                      00DD   280
                      00DD   281         ASSUME  DAP$K_ANY EQ 0
                      00DD   282         ASSUME  DAP$K_CYL EQ XAB$C_CYL
                      00DD   283         ASSUME  DAP$K_LBN EQ XAB$C_LBN
                      00DD   284         ASSUME  DAP$K_VBN EQ XAB$C_VBN
                      00DD   285
    83   09 A7    90  00DD   286         MOVB    XAB$B_ALN(R7),(R3)+      ; Store alignment options field
    51   08 A7    9A  00E1   287 20$:    MOVZBL  XAB$B_AOP(R7),R1         ; Get AOP bits returned by RMS
              52  D4  00E5   288         CLRL    R2                       ; Clear corresponding DAP bits
                      00E7   289         $MAPBIT XAB$V_CBT,DAP$V_CBT2     ; Map CBT bit
                      00EF   290         $MAPBIT XAB$V_CTG,DAP$V_CTG2     ; Map CTG bit
    10 69    34   E1  00F7   291         BBC     #DAP$V_VAXVMS,(R9),30$   ; Branch if partner is not VAX/VMS
                      00FB   292         $MAPBIT XAB$V_HRD,DAP$V_HRD      ; Map HRD bit
                      0103   293         $MAPBIT XAB$V_ONC,DAP$V_ONC      ; Map ONC bit
       51    52   D0  010B   294 30$:    MOVL    R2,R1                    ; Move data to correct register
           FEEF'  30  010E   295         BSBW    FAL$CVT_BN4_EXT          ; Store AOP as an extensible field
                      0111   296
                      0111   297 ;
                      0111   298 ; Include the LOC, ALQ, AID, BKZ, and DEQ fields in the message.
                      0111   299 ;
                      0111   300
    07 69    34   E1  0111   301         BBC     #DAP$V_VAXVMS,(R9),40$   ; Branch if partner is not VAX/VMS
    51   0C A7    D0  0115   302         MOVL    XAB$L_LOC(R7),R1         ; Get starting location value
           FEE4'  30  0119   303         BSBW    FAL$CVT_BN4_IMG          ; Store LOC as an image field
    51   10 A7    D0  011C   304 40$:    MOVL    XAB$L_ALQ(R7),R1         ; Get allocation quantity value
           FEDD'  30  0120   305         BSBW    FAL$CVT_BN4_IMG          ; Store ALQ as an image field
    83   17 A7    90  0123   306         MOVB    XAB$B_AID(R7),(R3)+      ; Store area identification field
    83   16 A7    90  0127   307         MOVB    XAB$B_BKZ(R7),(R3)+      ; Store bucket size field
    83   14 A7    B0  012B   308         MOVW    XAB$W_DEQ(R7),(R3)+      ; Store default extension quantity field
           FECE'  30  012F   309         BSBW    FAL$BUILD_TAIL           ; Finish building message
              05      0132   310         RSB                              ; Exit
```

```
                                 0133   312             .SBTTL  FAL$ENCODE_SUM
                             00000133   313             .PSECT  FAL$CODE        NOSHR,EXE,RD,NOWRT,BYTE
                                 0133   314
                                 0133   315     ;++
                                 0133   316     ; Functional Description:
                                 0133   317     ;
                                 0133   318     ;       FAL$ENCODE_SUM builds the DAP Summary message.
                                 0133   319     ;
                                 0133   320     ; Calling Sequence:
                                 0133   321     ;
                                 0133   322     ;       BSBW    FAL$ENCODE_SUM
                                 0133   323     ;
                                 0133   324     ; Input Parameters:
                                 0133   325     ;
                                 0133   326     ;       R8      Address of FAL work area
                                 0133   327     ;       R9      Address of DAP control block
                                 0133   328     ;       R10     Address of FAB
                                 0133   329     ;       R11     Address of RAB
                                 0133   330     ;
                                 0133   331     ; Implicit Inputs:
                                 0133   332     ;
                                 0133   333     ;       FAB$B_ORG
                                 0133   334     ;
                                 0133   335     ; Output Parameters:
                                 0133   336     ;
                                 0133   337     ;       R0-R6   Destroyed
                                 0133   338     ;       R7      Address of XAB
                                 0133   339     ;
                                 0133   340     ; Implicit Outputs:
                                 0133   341     ;
                                 0133   342     ;       None
                                 0133   343     ;
                                 0133   344     ; Completion Codes:
                                 0133   345     ;
                                 0133   346     ;       None
                                 0133   347     ;
                                 0133   348     ; Side Effects:
                                 0133   349     ;
                                 0133   350     ;       None
                                 0133   351     ;
                                 0133   352     ;--
                                 0133   353
                                 0133   354     FAL$ENCODE_SUM::                                ; Entry point
          57   03A4 C8     DE    0133   355             MOVAL   FAL$L_SUMXAB(R8),R7             ; Get address of Summary XAB
               50   0C     D0    0138   356             MOVL    #DAP$K_SUM_MSG,R0               ; Get message type value
                  FEC2'    30    013B   357             BSBW    FAL$BUILD_HEAD                  ; Construct message header
          20     1D AA     91    013E   358             CMPB    FAB$B_ORG(R10),#FAB$C_IDX ; Build dummy message (all fields
                     0F    12    0142   359             BNEQ    10$                             ;  defaulted) if file ORG is not IDX
                                 0144   360
                                 0144   361             ASSUME  DAP$V_NOK LT 7
                                 0144   362             ASSUME  DAP$V_NOA LT 7
                                 0144   363             ASSUME  DAP$V_PVN LT 7
                                 0144   364
               83   0B     90    0144   365             MOVB    #<DAP$M_NOK!-                   ; Get summary menu value
                                 0147   366                       DAP$M_NOA!-                   ;
                                 0147   367                       DAP$M_PVN!-                   ;
                                 0147   368                     0>,(R3)+                        ; Store sumenu as an extensible field
```

```
     83   09 A7   90  0147    369          MOVB    XAB$B_NOK(R7),(R3)+        ; Store number of keys field
     83   08 A7   90  014B    370          MOVB    XAB$B_NOA(R7),(R3)+        ; Store number of allocation areas field
     83   0A A7   B0  014F    371          MOVW    XAB$W_PVN(R7),(R3)+        ; Store prologue version number field
            FEAA' 30  0153    372 10$:     BSBW    FAL$BUILD_TAIL             ; Finish building message
                  05  0156    373          RSB                                ; Exit
```

```
                         0157          .SBTTL  FAL$ENCODE_TIM
                     00000157          .PSECT  FAL$CODE        NOSHR,EXE,RD,NOWRT,BYTE
                         0157
                         0157   ;++
                         0157   ; Functional Description:
                         0157   ;
                         0157   ;       FAL$ENCODE_TIM builds the DAP Date and Time message.
                         0157   ;
                         0157   ; Calling Sequence:
                         0157   ;
                         0157   ;       BSBW    FAL$ENCODE_TIM
                         0157   ;
                         0157   ; Input Parameters:
                         0157   ;
                         0157   ;       R8      Address of FAL work area
                         0157   ;       R9      Address of DAP control block
                         0157   ;       R10     Address of FAB
                         0157   ;       R11     Address of RAB
                         0157   ;
                         0157   ; Implicit Inputs:
                         0157   ;
                         0157   ;       DAP$V_GEQ_V60
                         0157   ;
                         0157   ; Output Parameters:
                         0157   ;
                         0157   ;       R0-R6   Destroyed
                         0157   ;       R7      Address of XAB
                         0157   ;
                         0157   ; Implicit Outputs:
                         0157   ;
                         0157   ;       None
                         0157   ;
                         0157   ; Completion Codes:
                         0157   ;
                         0157   ;       None
                         0157   ;
                         0157   ; Side Effects:
                         0157   ;
                         0157   ;       None
                         0157   ;
                         0157   ;--
                         0157
                         0157   FAL$ENCODE_TIM::                        ; Entry point
57   0320 C8   DE        0157          MOVAL   FAL$L_DATXAB(R8),R7     ; Get address of Date and Time XAB
     50   0D   D0        015C          MOVL    #DAP$K_TIM_MSG,R0       ; Get message type value
        FE9E'  30        015F          BSBW    FAL$BUILD_HEAD          ; Construct message header
                         0162
                         0162   ;
                         0162   ; Construct date and time menu value.
                         0162   ; Send only time fields that have a non-zero 64-bit time value as zero means
                         0162   ; the current date and time, not 17-NOV-1858! (actually only the upper 32-bits
                         0162   ; will be tested for zero, i.e., any time on 17-NOV-1858 will be considered
                         0162   ; as the default time.)
                         0162   ;
                         0162
                         0162          ASSUME  DAP$V_CDT EQ 0
                         0162          ASSUME  DAP$V_CDT+1 EQ DAP$V_RDT
```

```
                       0162    432             ASSUME  DAP$V_RDT+1 EQ DAP$V_EDT
                       0162    433             ASSUME  DAP$V_EDT+1 EQ DAP$V_RVN
                       0162    434             ASSUME  DAP$V_RVN+1 EQ DAP$V_BDT
                       0162    435
          54     D4    0162    436             CLRL    R4                          ; Initialize time menu field
       18 A7     D5    0164    437             TSTL    XAB$Q_CDT+4(R7)             ; Branch if creation date and time
          03     13    0167    438             BEQL    10$                         ;  is zero
     54   01     88    0169    439             BISB2   #DAP$M_CDT,R4               ; Otherwise, send field
       10 A7     D5    016C    440 10$:        TSTL    XAB$Q_RDT+4(R7)             ; Branch if revision date and time
          03     13    016F    441             BEQL    20$                         ;  is zero
     54   02     88    0171    442             BISB2   #DAP$M_RDT,R4               ; Otherwise, send field
       20 A7     D5    0174    443 20$:        TSTL    XAB$Q_EDT+4(R7)             ; Branch if expiration date and time
          03     13    0177    444             BEQL    30$                         ;  is zero
     54   04     88    0179    445             BISB2   #DAP$M_EDT,R4               ; Otherwise, send field
  08 69   25     E1    017C    446 30$:        BBC     #DAP$V_GEQ_V60,(R9),40$     ; Branch if partner uses DAP before V6.0
       28 A7     D5    0180    447             TSTL    XAB$Q_BDT+4(R7)             ; Branch if backup date and time
          03     13    0183    448             BEQL    40$                         ;  is zero
     54   10     88    0185    449             BISB2   #DAP$M_BDT,R4               ; Otherwise, send field
     54   08     88    0188    450 40$:        BISB2   #DAP$M_RVN,R4               ; Send revision number field
     83   54     90    018B    451             MOVB    R4,(R3)+                    ; Store TIMENU as an extensible field
                       018E    452
                       018E    453 ;
                       018E    454 ; Now process each time field.
                       018E    455 ;
                       018E    456
  06 54   00     E1    018E    457             BBC     #DAP$V_CDT,R4,50$           ; Branch if CDT is not to be included
  50   14 A7     7E    0192    458             MOVAQ   XAB$Q_CDT(R7),R0            ; Get address of 64-bit value for
                       0196    459                                                 ;  creation date and time
          26     10    0196    460             BSBB    CONVERT_TIME                ; Store CDT as an image field
  06 54   01     E1    0198    461 50$:        BBC     #DAP$V_RDT,R4,60$           ; Branch if RDT is not to be included
  50   0C A7     7E    019C    462             MOVAQ   XAB$Q_RDT(R7),R0            ; Get address of 64-bit value for
                       01A0    463                                                 ;  revision date and time
          1C     10    01A0    464             BSBB    CONVERT_TIME                ; Store RDT as an image field
  06 54   02     E1    01A2    465 60$:        BBC     #DAP$V_EDT,R4,70$           ; Branch if EDT is not to be included
  50   1C A7     7E    01A6    466             MOVAQ   XAB$Q_EDT(R7),R0            ; Get address of 64-bit value for
                       01AA    467                                                 ;  expiration date and time
          12     10    01AA    468             BSBB    CONVERT_TIME                ; Store EDT as an image field
  83   08 A7     B0    01AC    469 70$:        MOVW    XAB$W_RVN(R7),(R3)+         ; Store revision number field
  06 54   04     E1    01B0    470             BBC     #DAP$V_BDT,R4,80$           ; Branch if BDT is not to be included
  50   24 A7     7E    01B4    471             MOVAQ   XAB$Q_BDT(R7),R0            ; Get address of 64-bit value for
                       01B8    472                                                 ;  backup date and time
          04     10    01B8    473             BSBB    CONVERT_TIME                ; Store BDT as an image field
       FE43'     30    01BA    474 80$:        BSBW    FAL$BUILD_TAIL              ; Finish building message
                 05    01BD    475             RSB                                 ; Exit
                       01BE    476
                       01BE    477 ;
                       01BE    478 ; This routine converts a time value in 64-bit binary format to an ASCII string.
                       01BE    479 ; Then it stores the string as an 18-byte fixed length field in the DAP message
                       01BE    480 ; with the first two digits of the year removed (per DAP specification).
                       01BE    481 ;
                       01BE    482
                       01BE    483 CONVERT_TIME:                                   ; Entry point
       5E 20     C2    01BE    484             SUBL2   #<20+12>,SP                 ; Allocate space from the stack
       52 5E     D0    01C1    485             MOVL    SP,R2                       ; Save address of work area
  14 A2   14     D0    01C4    486             MOVL    #20,20(R2)                  ; Form descriptor of buffer to receive
  18 A2   5E     D0    01C8    487             MOVL    SP,24(R2)                   ;  ASCII time string
                       01CC    488             $ASCTIM_S-                          ; Convert binary time to ASCII time
```

```
                     01CC                    TIMLEN=28(R2)-          ;  Address of word to return string size
                     01CC                    TIMBUF=20(R2)-          ;  Address of descriptor for buffer
                     01CC                    TIMADR=(R0)-            ;  Address of 64-bit time value
                     01CC                    CVTFLG=#0               ;  Flag set to request date and time
                     01DD            $CHECK_SS                       ; Check status code and exit on failure
        62   20   91 01E0            CMPB    #^A\ \,(R2)             ; Convert leading space to zero in
             03   12 01E3            BNEQ    10$                     ;  day-of-month field to conform to
        62   30   90 01E5            MOVB    #^A\0\,(R2)             ;  the DAP V6.0 specification
                     01E8                                            ; Store time field omitting the two
                     01E8                                            ;  century digits
             10   BB 01E8  10$:      PUSHR   #^M<R4>                 ; Save time menu mask
     63 62   07   28 01EA            MOVC3   #7,(R2),(R3)            ; Copy bytes 1-7 of input string
63   02 A1   0B   28 01EE            MOVC3   #11,2(R1),(R3)          ; Copy bytes 9-20 of input string
             10   BA 01F3            POPR    #^M<R4>                 ; Restore time menu mask
        5E   20   C0 01F5            ADDL2   #<20+12>,SP             ; Deallocate space from the stack
                  05 01F8            RSB                             ; Exit
                  01F9          .SBTTL  FAL$ENCODE_PRO
              000001F9          .PSECT  FAL$CODE            NOSHR,EXE,RD,NOWRT,BYTE
                  01F9
                  01F9   ;++
                  01F9   ; Functional Description:
                  01F9   ;
                  01F9   ;      FAL$ENCODE_PRO builds the DAP Protection message.
                  01F9   ;
                  01F9   ; Calling Sequence:
                  01F9   ;
                  01F9   ;      BSBW    FAL$ENCODE_PRO
                  01F9   ;
                  01F9   ; Input Parameters:
                  01F9   ;
                  01F9   ;      R8      Address of FAL work area
                  01F9   ;      R9      Address of DAP control block
                  01F9   ;      R10     Address of FAB
                  01F9   ;      R11     Address of RAB
                  01F9   ;
                  01F9   ; Implicit Inputs:
                  01F9   ;
                  01F9   ;      None
                  01F9   ;
                  01F9   ; Output Parameters:
                  01F9   ;
                  01F9   ;      R0-R6   Destroyed
                  01F9   ;      R7      Address of XAB
                  01F9   ;
                  01F9   ; Implicit Outputs:
                  01F9   ;
                  01F9   ;      None
                  01F9   ;
                  01F9   ; Completion Codes:
                  01F9   ;
                  01F9   ;      None
                  01F9   ;
                  01F9   ; Side Effects:
                  01F9   ;
                  01F9   ;      None
                  01F9   ;
                  01F9   ;--
                  01F9
                  01F9   FAL$ENCODE_PRO::                         ; Entry point
57  034C C8   DE  01F9          MOVAL   FAL$L_PROXAB(R8),R7      ; Get address of Protection XAB
50    0E      D0  01FE          MOVL    #DAP$K_PRO_MSG,R0        ; Get message type value
      FDFC'   30  0201          BSBW    FAL$BUILD_HEAD           ; Construct message header
                  0204
                  0204          ASSUME  DAP$V_OWNER LT 7
                  0204          ASSUME  DAP$V_PROSYS LT 7
                  0204          ASSUME  DAP$V_PROOWN LT 7
                  0204          ASSUME  DAP$V_PROGRP LT 7
                  0204          ASSUME  DAP$V_PROWLD LT 7
                  0204
83    1F      90  0204          MOVB    #<DAP$M_OWNER!-          ; Get protection menu value
                  0207                    DAP$M_PROSYS!-         ;
                  0207                    DAP$M_PROOWN!-         ;
                  0207                    DAP$M_PROGRP!-         ;
```

```
                        0207                     DAP$M_PROWLD!-            ;
                        0207                   0>,(R3)+                    ; Store PROMENU as an extensible field
                        0207
                        0207           ;
                        0207           ; Include the OWNER field in the message.
                        0207           ;
                        0207
          5E    1C  C2  0207                SUBL2   #<16+12>,SP            ; Allocate space from the stack
          52    5E  D0  020A                MOVL    SP,R2                  ; Save address of work area
       10 A2    10  D0  020D                MOVL    #16,16(R2)             ; Form descriptor of buffer to receive
       14 A2    5E  D0  0211                MOVL    SP,20(R2)              ;  ASCII string
       50    0E A7  3C  0215                MOVZWL  XAB$W_GRP(R7),R0       ; Get group UIC value
       51    0C A7  3C  0219                MOVZWL  XAB$W_MBM(R7),R1       ; Get member UIC value
                        021D                $FAO_S-                        ; Format the UIC string
                        021D                        CTRSTR=W^FAL$GQ_UIC-   ;  Address of FAO control string
                        021D                        OUTLEN=24(R2)-         ;  Address of receive string length
                        021D                        OUTBUF=16(R2)-         ;  Address of buffer descriptor
                        021D                        P1=R0-                 ;  Group number of file owner
                        021D                        P2=R1                  ;  Member number of file owner
                        0232                $CHECK_SS                      ; Check status code and exit on failure
       50    18 A2  3C  0235                MOVZWL  24(R2),R0              ; Get length of returned string
          83    50  90  0239                MOVB    R0,(R3)+               ; Store owner as an image field
    63    62    50  28  023C                MOVC3   R0,(R2),(R3)           ; Copy owner string to message
          5E    1C  C0  0240                ADDL2   #<16+12>,SP            ; Deallocate space from the stack
                        0243
                        0243           ;
                        0243           ; Construct the four protection fields: PROSYS, PROOWN, PROGRP, and PROWLD.
                        0243           ;
                        0243
                        0243                ASSUME  DAP$V_RED_ACC EQ XAB$V_NOREAD
                        0243                ASSUME  DAP$V_WRT_ACC EQ XAB$V_NOWRITE
                        0243                ASSUME  DAP$V_EXE_ACC EQ XAB$V_NOEXE
                        0243                ASSUME  DAP$V_DLT_ACC EQ XAB$V_NODEL
                        0243
                        0243                ASSUME  DAP$V_RED_ACC LT 7
                        0243                ASSUME  DAP$V_WRT_ACC LT 7
                        0243                ASSUME  DAP$V_EXE_ACC LT 7
                        0243                ASSUME  DAP$V_DLT_ACC LT 7
                        0243
         50  08 A7  3C  0243                MOVZWL  XAB$W_PRO(R7),R0       ; Get protection value
 51   50  04    00  EF  0247                EXTZV   #XAB$V_SYS,#4,R0,R1    ; Store system protection field
          83    51  90  024C                MOVB    R1,(R3)+               ;  as an extensible field
 51   50  04    04  EF  024F                EXTZV   #XAB$V_OWN,#4,R0,R1    ; Store owner protection field
          83    51  90  0254                MOVB    R1,(R3)+               ;  as an extensible field
 51   50  04    08  EF  0257                EXTZV   #XAB$V_GRP,#4,R0,R1    ; Store group protection field
          83    51  90  025C                MOVB    R1,(R3)+               ;  as an extensible field
 51   50  04    0C  EF  025F                EXTZV   #XAB$V_WLD,#4,R0,R1    ; Store world protection field
          83    51  90  0264                MOVB    R1,(R3)+               ;  as an extensible field
             FD96'  30  0267                BSBW    FAL$BUILD_TAIL         ; Finish building message
                    05  026A                RSB                            ; Exit
                        026B
                        026B                .END                           ; End of module
```

```
$$T2                          = 00000005               DAP$L_CMWA                           00000030
CONVERT_TIME                    000001BE R      02     DAP$L_CRC_RSLT                       00000020
DAP$B_AID                       00000050               DAP$L_DCODE_STS                      00000018
DAP$B_ALN                       00000044               DAP$L_DEV                            00000068
DAP$B_AOP                       00000045               DAP$L_DVB                            00000078
DAP$B_BITCNT                    00000035               DAP$L_EBK                            00000078
DAP$B_BKS                       00000050               DAP$L_FOP1                           00000064
DAP$B_BKZ                       00000051               DAP$L_HBK                            00000074
DAP$B_BSZ                       00000052               DAP$L_KEYMENU                        00000040
DAP$B_DAN                       00000070               DAP$L_LOC                            00000048
DAP$B_DATATYPE                  00000044               DAP$L_MRN                            00000058
DAP$B_DBS                       0000007C               DAP$L_MSG_MASK                       0000001C
DAP$B_DCODE_FID                 00000019               DAP$L_RVB                            00000074
DAP$B_DCODE_MAC                 0000001B               DAP$L_SBN                            0000007C
DAP$B_DCODE_MSG                 0000001A               DAP$L_SSPWA                          00000080
DAP$B_DTP                       00000071               DAP$L_TEMP                           00000090
DAP$B_FLAGS                     00000031               DAP$M_AID                          = 00000040
DAP$B_FLG                       00000048               DAP$M_ALN                          = 00000002
DAP$B_FSZ                       00000051               DAP$M_ALQ2                         = 00000020
DAP$B_IAN                       0000006E               DAP$M_AOP                          = 00000004
DAP$B_IBS                       0000007D               DAP$M_BDT                          = 00000010
DAP$B_LAN                       0000006F               DAP$M_BITCNT                       = 00000008
DAP$B_LEN256                    00000034               DAP$M_BKZ                          = 00000080
DAP$B_LENGTH                    00000033               DAP$M_CDT                          = 00000001
DAP$B_LVL                       0000007E               DAP$M_CMPFMT                       = 00000008
DAP$B_NOA                       00000045               DAP$M_DAN                          = 00000200
DAP$B_NOK                       00000044               DAP$M_DBS                          = 00004000
DAP$B_NOR                       00000046               DAP$M_DEQ2                         = 00000100
DAP$B_NSG                       00000049               DAP$M_DFL                          = 00000002
DAP$B_NUL                       0000006D               DAP$M_DMO                          = 00002000
DAP$B_ORG                       00000045               DAP$M_DTP                          = 00000400
DAP$B_RAT                       00000047               DAP$M_DVB                          = 00002000
DAP$B_REF                       0000006C               DAP$M_EDT                          = 00000004
DAP$B_RFM                       00000046               DAP$M_EMBEDDED                     = 00000010
DAP$B_SIZ                       0000005C               DAP$M_FLG                          = 00000001
DAP$B_SIZ_TMP                   0000004A               DAP$M_IAN                          = 00000080
DAP$B_STREAMID                  00000032               DAP$M_IBS                          = 00008000
DAP$B_TKS                       0000007F               DAP$M_IFL                          = 00000004
DAP$B_TYPE                      00000030               DAP$M_IMAGE                        = 00000002
DAP$B_X_FIELD                   00000024               DAP$M_KNM                          = 00000020
DAP$C_BLN                       000000C0               DAP$M_LAN                          = 00000100
DAP$K_ALL_MSG                 = 0000000B               DAP$M_LOC                          = 00000008
DAP$K_ANY                     = 00000000               DAP$M_LSA                          = 00000040
DAP$K_BLN                       000000C0               DAP$M_LVL                          = 00010000
DAP$K_CYL                     = 00000001               DAP$M_MACY11                       = 00000080
DAP$K_FIX                     = 00000001               DAP$M_MRL                          = 00040000
DAP$K_KEY_MSG                 = 0000000A               DAP$M_NOA                          = 00000002
DAP$K_LBN                     = 00000002               DAP$M_NOK                          = 00000001
DAP$K_PRO_MSG                 = 0000000E               DAP$M_NSG                          = 00000008
DAP$K_SEQ                     = 00000000               DAP$M_NUL                          = 00000040
DAP$K_STG                     = 00000000               DAP$M_OWNER                        = 00000001
DAP$K_SUM_MSG                 = 0000000C               DAP$M_PROGRP                       = 00000008
DAP$K_TIM_MSG                 = 0000000D               DAP$M_PROOWN                       = 00000004
DAP$K_VBN                     = 00000003               DAP$M_PROSYS                       = 00000002
DAP$L_ALQ1                      0000004C               DAP$M_PROWLD                       = 00000010
DAP$L_ALQ2                      0000004C               DAP$M_PVN                          = 00000008
DAP$L_ATTMENU                   00000040               DAP$M_RDT                          = 00000002
```

```
DAP$M_REF                    = 00000010          DAP$W_IFL                        00000046
DAP$M_RVB                    = 00000800          DAP$W_LRL                        00000070
DAP$M_RVN                    = 00000008          DAP$W_MRL                        00000072
DAP$M_SEGMENT                = 00000040          DAP$W_MRS                        0000004A
DAP$M_TKS                    = 00020000          DAP$W_PARTNER                    00000006
DAP$M_TMP1$                  = 0000FE00          DAP$W_POS                        0000004C
DAP$M_TMP2$                  = 0000FE00          DAP$W_POS_TMP                    0000004A
DAP$M_TMP3$                  = 00020000          DAP$W_PROGRP                     00000054
DAP$M_TMP4$                  = 01000000          DAP$W_PROMENU                    00000040
DAP$M_TMP5$                  = F0000000          DAP$W_PROOWN                     00000052
DAP$M_VOL                    = 00000001          DAP$W_PROSYS                     00000050
DAP$M_ZERO                   = 00000080          DAP$W_PROWLD                     00000056
DAP$Q_ADT                      00000070          DAP$W_PVN                        00000042
DAP$Q_BDT                      00000060          DAP$W_RVN                        00000042
DAP$Q_CDT                      00000048          DAP$W_SUMENU                     00000040
DAP$Q_DCODE_FLG                00000000          DAP$W_TIMENU                     00000040
DAP$Q_EDT                      00000058          DAP$W_VERSION                    00000004
DAP$Q_KNM                      00000064          DAP$W_VOL                        00000042
DAP$Q_MSG_BUF1                 00000008          FAB$B_ORG                      = 0000001D
DAP$Q_MSG_BUF2                 00000010          FAB$C_IDX                      = 00000020
DAP$Q_OWNER                    00000048          FAL$BUILD_HEAD                   ********   X    02
DAP$Q_PDT                      00000068          FAL$BUILD_TAIL                   ********   X    02
DAP$Q_RDT                      00000050          FAL$B_ACCFUNC                    000001F6
DAP$Q_RUNSYS                   0000005C          FAL$B_ACCOPT                     000001F5
DAP$Q_SYSPEC                   00000038          FAL$B_DATATYPE                   000001F4
DAP$V_BDT                    = 00000004          FAL$B_DISABLE                    00000006
DAP$V_CBT2                   = 00000002          FAL$B_ENABLE                     00000005
DAP$V_CDT                    = 00000000          FAL$B_LOGGING                    00000004
DAP$V_CHG                    = 00000001          FAL$B_MISCOPT                    00000007
DAP$V_CTG2                   = 00000001          FAL$B_RAC                        000001F7
DAP$V_DLT_ACC                = 00000003          FAL$B_RBK_CACHE                  00000012
DAP$V_DUP                    = 00000000          FAL$B_RCVBUFIDX                  00000011
DAP$V_EDT                    = 00000002          FAL$B_VALUE                      00000010
DAP$V_EXE_ACC                = 00000002          FAL$CHECK_SS                     ********   X    02
DAP$V_GEQ_V60                = 00000025          FAL$CVT_BN4_EXT                  ********   X    02
DAP$V_HRD                    = 00000000          FAL$CVT_BN4_IMG                  ********   X    02
DAP$V_NOA                    = 00000001          FAL$C_WRKBLN                     00002000
DAP$V_NOK                    = 00000000          FAL$ENCODE_ALL                   000000B7 RG    02
DAP$V_NUL_CHR                = 00000002          FAL$ENCODE_KEY                   00000000 RG    02
DAP$V_ONC                    = 00000003          FAL$ENCODE_PRO                   000001F9 RG    02
DAP$V_OWNER                  = 00000000          FAL$ENCODE_SUM                   00000133 RG    02
DAP$V_PROGRP                 = 00000003          FAL$ENCODE_TIM                   00000157 RG    02
DAP$V_PROOWN                 = 00000002          FAL$GQ_UIC                       ********   X    02
DAP$V_PROSYS                 = 00000001          FAL$K_ALLXAB                   = 00000020
DAP$V_PROWLD                 = 00000004          FAL$K_KEYXAB                   = 0000004C
DAP$V_PVN                    = 00000003          FAL$K_WRKBLN                     00002000
DAP$V_RDT                    = 00000001          FAL$L_ALLXAB                     00000C00
DAP$V_RED_ACC                = 00000000          FAL$L_ALLXABINI                  00000074
DAP$V_RVN                    = 00000003          FAL$L_CHAIN_NXT                  0000007C
DAP$V_VAXVMS                 = 00000034          FAL$L_DATXAB                     00000320
DAP$V_WRT_ACC                = 00000001          FAL$L_FAB                        00000200
DAP$W_ALLMENU                  00000040          FAL$L_FAB2                       00000800
DAP$W_BLS                      00000048          FAL$L_FHCXAB                     000002F4
DAP$W_DEQ1                     00000054          FAL$L_FOP                        000001F8
DAP$W_DEQ2                     00000052          FAL$L_KEYNAM                     00001C00
DAP$W_DFL                      00000044          FAL$L_KEYXAB                     00001000
DAP$W_FFB                      00000072          FAL$L_KEYXABINI                  00000078
```

```
FAL$L_NAM             00000294              XAB$B_BKZ           = 00000016
FAL$L_NAM2            00000850              XAB$B_DAN           = 0000000A
FAL$L_NUMBER          000001FC              XAB$B_DBS           = 0000000D
FAL$L_PROXAB          0000034C              XAB$B_DTP           = 00000013
FAL$L_RAB             00000250              XAB$B_FLG           = 00000012
FAL$L_RCVBUF          0000005C              XAB$B_IAN           = 00000008
FAL$L_RDTXAB          000003B0              XAB$B_IBS           = 0000000C
FAL$L_RMS_PTR         0000006C              XAB$B_LAN           = 00000009
FAL$L_STB             000000C0              XAB$B_LVL           = 0000000B
FAL$L_SUMXAB          000003A4              XAB$B_NOA           = 00000008
FAL$L_TEMP            000003F4              XAB$B_NOK           = 00000009
FAL$L_USE_SC1         000000A8              XAB$B_NSG           = 00000014
FAL$L_USE_SC2         000000AC              XAB$B_NUL           = 00000015
FAL$L_USE_VER         000000A4              XAB$B_REF           = 00000017
FAL$Q_BLD             00000050              XAB$B_SIZ           = 0000002E
FAL$Q_DIRNAME         00000088              XAB$B_TKS           = 00000016
FAL$Q_FALLOG          00000090              XAB$C_CYL           = 00000001
FAL$Q_FLG             00000000              XAB$C_LBN           = 00000002
FAL$Q_MBX             00000038              XAB$C_VBN           = 00000003
FAL$Q_MBXIOSB         00000030              XAB$L_ALQ           = 00000010
FAL$Q_RCV             00000040              XAB$L_DVB           = 0000003C
FAL$Q_RCVIOSB         00000020              XAB$L_KNM           = 00000038
FAL$Q_RMS             00000064              XAB$L_LOC           = 0000000C
FAL$Q_STATE_CTX       00000008              XAB$L_RVB           = 0000000E
FAL$Q_SYSNET          00000098              XAB$Q_BDT           = 00000024
FAL$Q_TEMP            000003F8              XAB$Q_CDT           = 00000014
FAL$Q_VOLNAME         00000080              XAB$Q_EDT           = 0000001C
FAL$Q_XMT             00000048              XAB$Q_RDT           = 0000000C
FAL$Q_XMTIOSB         00000028              XAB$V_CBT           = 00000005
FAL$T_DAP             00000100              XAB$V_CHG           = 00000001
FAL$T_DIRNAME         00001F00              XAB$V_CTG           = 00000007
FAL$T_EXPAND          00000500              XAB$V_DUP           = 00000000
FAL$T_EXPAND2         00000A00              XAB$V_GRP           = 00000008
FAL$T_FALLOG          00001C00              XAB$V_HRD           = 00000000
FAL$T_FILESPEC        00000400              XAB$V_NODEL         = 00000003
FAL$T_FILESPEC2       00000900              XAB$V_NOEXE         = 00000002
FAL$T_KEYBUF          00000700              XAB$V_NOREAD        = 00000000
FAL$T_MBXBUF          00001980              XAB$V_NOWRITE       = 00000001
FAL$T_PRTBUF1         00001A00              XAB$V_NUL           = 00000002
FAL$T_PRTBUF2         00001B00              XAB$V_ONC           = 00000001
FAL$T_RESULT          00000600              XAB$V_OWN           = 00000004
FAL$T_RESULT2         00000B00              XAB$V_SYS           = 00000000
FAL$T_SYSNET          00001D00              XAB$V_WLD           = 0000000C
FAL$T_VOLNAME         00001E00              XAB$W_DEQ           = 00000014
FAL$W_DAPBUFSIZ       0000001A              XAB$W_DFL           = 0000001C
FAL$W_DISPLAY         00000070              XAB$W_GRP           = 0000000E
FAL$W_LNKCHN          0000001C              XAB$W_IFL           = 0000001A
FAL$W_MBXCHN          0000001E              XAB$W_MBM           = 0000000C
FAL$W_QIOBUFSIZ       00000018              XAB$W_MRL           = 00000018
FAL$W_RECEIVED        00000072              XAB$W_POS           = 0000001E
FAL$W_USE_DBS         000000A0              XAB$W_PRO           = 00000008
FAL$W_USE_SYS         000000A2              XAB$W_PVN           = 0000000A
SYS$ASCTIM            ********  GX    02    XAB$W_RVN           = 00000008
SYS$FAO               ********   X    02    XAB$W_VOL           = 0000000A
XAB$B_AID           = 00000017
XAB$B_ALN           = 00000009
XAB$B_AOP           = 00000008
```

```
                                                     +----------------+
                                                     ! Psect synopsis !
                                                     +----------------+

PSECT name                      Allocation          PSECT No.  Attributes
----------                      ----------          ---------  ----------
.  ABS  .                       00000000  (      0.)  00 (  0.)  NOPIC   USR   CON   ABS   LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
$ABS$                           00002000  (   8192.)  01 (  1.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD     WRT NOVEC BYTE
FAL$CODE                        0000026B  (    619.)  02 (  2.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD   NOWRT NOVEC BYTE

                                                  +------------------------+
                                                  ! Performance indicators !
                                                  +------------------------+

Phase                        Page faults    CPU Time       Elapsed Time
-----                        -----------    --------       ------------
Initialization                        35    00:00:00.04    00:00:01.07
Command processing                   139    00:00:00.41    00:00:03.29
Pass 1                               342    00:00:09.19    00:00:31.10
Symbol table sort                      0    00:00:01.02    00:00:05.62
Pass 2                               117    00:00:01.80    00:00:06.69
Symbol table output                   47    00:00:00.19    00:00:01.55
Psect synopsis output                  2    00:00:00.02    00:00:00.02
Cross-reference output                 0    00:00:00.00    00:00:00.00
Assembler run totals                 684    00:00:12.67    00:00:49.34

The working set limit was 1650 pages.
72669 bytes (142 pages) of virtual memory were used to buffer the intermediate code.
There were 60 pages of symbol table space allocated to hold 1145 non-local and 26 local symbols.
614 source lines were read in Pass 1, producing 15 object records in Pass 2.
30 pages of virtual memory were used to define 28 macros.

                                                 +--------------------------+
                                                 ! Macro library statistics !
                                                 +--------------------------+

Macro library name                           Macros defined
------------------                           --------------
_$255$DUA28:[FAL.OBJ]FAL.MLB;1                         11
_$255$DUA28:[SYSLIB]STARLET.MLB;2                      14
TOTALS (all libraries)                                 25

1500 GETS were required to define 25 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:FALBLDXAB/OBJ=OBJ$:FALBLDXAB MSRC$:FALBLDXAB/UPDATE=(ENH$:FALBLDXAB)+LIB$:FAL/LIB
```

0174 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY